A Bluetooth-enabled HiperLAN/2 receiver

Roel Schiphorst, Fokke W. Hoeksema and Cornelis H. Slump
University of Twente, Department of Electrical
Engineering, Mathematics and Computer Science (EEMCS),
Signals and Systems group,
P.O. box 217 - 7500 AE Enschede - The Netherlands
Phone: +31 53 489 2770 Fax: +31 53 489 1060
Email: {r.schiphorst, f.w.hoeksema, c.h.slump}@utwente.nl

***Abstract*— In our SDR project we aim to combine a GFSK receiver (Bluetooth) with an OFDM receiver (HiperLAN/2). Other WLAN standards use the same frequency bands and modulation techniques. So our Bluetooth-enabled HiperLAN/2 receiver can easily be adapted to other WLAN standards.**

This paper focusses on the integration of the two receivers. A functional architecture of a combined receiver and its computational requirements will be presented.

I. Introduction

In our Software-Defined Radio (SDR) project [1] we integrate two different types of wireless LAN (WLAN) standards, HiperLAN/2 and Bluetooth on one common hardware platform. The focus of our project is on designing the front end of a receiver (from antenna to demodulation in bits) of an SDR system for a mobile terminal.

Although dedicated receivers (for one standard) will always consume less power (a factor 10 or more), SDR has several advantages for both consumers and manufacturers. For manufacturers this could result in shorter development time and cheaper production due to higher volumes. Furthermore, SDR has advantages for consumers because it enables to provide new functionality by software updates without the need for new hardware.

Moreover, in digital communication the trend is, due to Moore's law, that more functionality of the radio transceiver is implemented digital, because the analog part of the transceiver remains the same in every fabrication technology whereas the digital part is scaled down. So the transceiver is more and more digitized. All these reasons enable software (defined) radio.

A. Outline

The outline of this paper is as follows. First an introduction is given on software (defined) radio and the SDR project at the University of Twente. Then the two receivers for both standards (HiperLAN/2 and Bluetooth) are discussed and their computational requirements will be presented. Commonly used Bluetooth receivers have a structure that differs considerably from a HiperLAN/2 receiver. So integration of the two receiver is rather difficult. By using a Maximum A posteriori Probability (MAP) receiver for Bluetooth, integration of both standards is more easily. Finally, a functional architecture of a combined receiver and conclusions are drawn.

B. Software radio

The abundance of digital communication standards in not only disadvantageous for consumers but also for manufacturers because they have to develop a new product for each standard. It is for that reason that the software-radio concept is emerging as a potential pragmatic solution: a software implementation of the user terminal able to dynamically adapt to the radio environment in which the terminal is located [2], [3]. For manufacturers this could result in shorter development time, cheaper production due to higher volumes. Furthermore SDR has advantages for consumers because it enables only software updates for new functionality without new hardware.

Because of the analog nature of the air interface, a software radio will always have an analog front end. In an ideal software radio, the analog-to-digital converter (ADC) and the digital-to-analog converter (DAC) are positioned directly after the antenna. Such an implementation is not feasible due to the power that such device would consume and other physical limitations [4]. It is therefore a challenge to design a system that preserves most properties of the ideal software radio while being realizable with current-day technology. Such a system is called a software-defined radio (SDR).

C. The Bluetooth-enabled HiperLAN/2 receiver project

In our Software-Defined-Radio (SDR) project [1] we aim to combine an instance of a GFSK receiver (Bluetooth) with a OFDM receiver (HiperLAN/2). Other WLAN standards use the same frequency bands and modulation techniques. Our focus is on the physical layer of the receiver: from antenna output to raw bits. The research is carried out by two chairs of the University of Twente: the IC-Design group which focusses on the analog part and the Signals and Systems group focussing on the digital part.

The vehicle of our project is a notebook to which we add the SDR functionality. This has three advantages. First, we can use the processing capabilities of the general purpose processor for digital signal processing. Second, in comparison to SDR for mobile phones, our demonstrator can consume much more power (in the order of 1 W). Third, a notebook is very suited for demonstration purposes.

Table I shows some characteristics of the physical layer of both standards. HiperLAN/2 is a high-speed Wireless LAN (WLAN) standard using Orthogonal Frequency Division

0-7803-7954-3/03/$17.00 ©2003 IEEE.

parameter	Bluetooth	HiperLAN/2
band	$2.4 - 2.48$ GHz	$5.15 - 5.725$ GHz
ch. spacing	1 MHz	20 MHz
modulation	GFSK	OFDM + BPSK/ QPSK/16-QAM/64-QAM
nom. bitrate (no FEC)	$172.8 - 723.2$ kbit/s	$12 - 72$ Mbit/s

TABLE I
SOME PHYSICAL LAYER CHARACTERISTICS OF BLUETOOTH AND HIPERLAN/2

Multiplexing (OFDM). Its physical layer is very similar to the 802.11a standard. Bluetooth on the other hand is a low cost, low speed standard, designed for replacing fixed cables. Bluetooth uses Gaussian Frequency Shift Keying (GFSK) which is also used by other standards such as IEEE 802.11b and DECT.

For our project we interpret SDR as an implementation technology: the HiperLAN/2 hardware is that complex to the Bluetooth hardware that the Bluetooth receiver may be added to the HiperLAN/2 at limited costs. This point of view on software radio differs from the views in [2] and [3]; flexible, universal, radio systems at each layer of the OSI model where manufacturers, network operators and consumer can benefit from this flexibility. Our interpretation on SDR is more focussed on the physical layer; an implementation technology, invisible for consumers, which enables shorter development time, patchability and cheaper production due to higher volumes for manufacturers.

In order to gain knowledge about Bluetooth and HiperLAN/2 receivers we first built a test-bed with two separate receivers [5]. The functional architecture is depicted in Fig. 1.

1) Analog front-end: A block schematic of the current implementation of the analog front end is given in Fig. 2 [6]. For HiperLAN/2 this is a zero-IF structure with an output of 1 channel at baseband. A HiperLAN/2 channel has a (complex) bandwidth of 20 MHz[1]. Analog-to-digital conversion has to be performed in quadrature and with a minimal sample rate of 20 MSPS (Million Samples Per Second). For our demonstrator we choose to use 80 MSPS, because power-efficient analog filters do not have a small transition band. The analog front end uses 3^{th} order Butterworth filters with a cut-off frequency of about 11 MHz. So part of the channel selection is performed digitally. In Bluetooth mode, the output of the analog front-end is also a 20 MHz wide signal containing 20 Bluetooth channels (a Bluetooth channel has a bandwidth of 1 MHz). As neighboring Bluetooth channels can be 40 dB stronger than the wanted channel [7], the ADC resolution should be at least 10 bit. (HiperLAN/2 has less stringent requirements.) In our project we use 12-bit ADCs.

2) Digital front-end: This test-bed has enabled us to estimate the processing power for both receivers (roughly 1.5 billion additions/multiplications for each receiver). As our project focusses on a daughter card for a computer/notebook, we can use the processing power of the notebooks' CPU. Current processors, such as the Pentium IV, posses special signal-processing instructions (MMX/SSE/SSE2) and have therefore huge processing capabilities. A Pentium IV can, for example, do four floating point multiplications during one clock cycle. So we expect that a large part of the receiver can be run real-time on a Pentium IV and current research focusses on this real-time demonstrator.

[1]So the bandwidth per quadrature channel is 10 MHz.

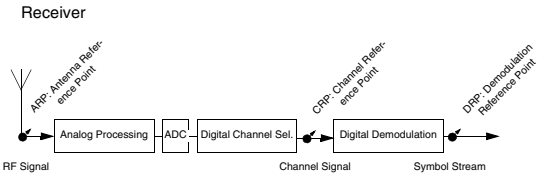

Fig. 1. Functional architecture of the SDR test bed

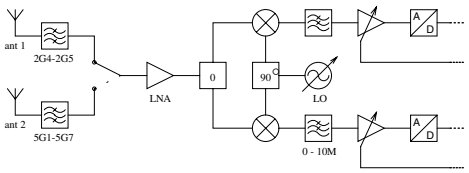

Fig. 2. Block schematic of the SDR receiver (analog part)

II. FUNCTIONAL ARCHITECTURE OF A HIPERLAN/2 RECEIVER

HiperLAN/2 uses OFDM, a multi-carrier modulation technique. Both the transmitter and receiver operate at 20 Million Samples Per Second (MSPS). An OFDM symbol has a duration of 4μs (80 complex samples) with 48 data and 4 pilot carriers. A MAC frame has a maximal duration of 2 ms [8]. This frame consists of 5 parts. For estimating computational requirements, we assume that all parts have equal duration and that we have to demodulate 2 parts (one common and one user part). These part have a duration of $\frac{2000}{5} * 2 = 800$ μs (200 OFDM symbols). Moreover we assume continuous transmission. (Of course, in realistic situations this is not the case.) The number of transmitted OFDM symbols per second is in our case $N_s = \frac{1}{T}M = \frac{1}{0.002} * 200 = 100000$ symbols. (M is the number of OFDM symbols in a burst that have to be demodulated and T the duration of a burst.) For all parts except the FFT, we assume that 16-bit fixed point calculations are sufficient.

The digital part of the HiperLAN/2 receiver consists of two parts; channel selection and demodulation (See Fig. 1).

A. Channel selection

For HiperLAN/2 receiver we use a zero-IF structure with an output of 1 channel at baseband. The output of the ADCs is a complex (2x12-bit) digital signal. As the analog low pass filter removes only non-adjacent channels, the digital channel selection has to remove the adjacent channels and reduce the sample rate to 20 MSPS. Initial simulations showed that 25-tap FIR filters is sufficient.

1) Computational requirements: The used ADCs are 12-bit, therefore 16-bit registers in the FIR filter should be enough. Assuming symmetric FIR filters operating at 80 MSPS (at the input rate), 25 taps and decimation factor 4, the computational load is: $2 * \frac{25}{2} * \frac{80}{4} = 500$ million 16-bit multiplications and $2 * (25 - 1) * 20 = 960$ million 16-bit additions per second.

B. Demodulation

A general receiver structure for OFDM is depicted in Fig. 3. To eliminate Inter Symbol Interference (ISI), each OFDM symbol contains a so called *prefix* (of 16 samples) that is a copy of the last part of the symbol. A MAC frame starts with special, known, symbols, so called *preambles*. The synchronization part can use these preambles to detect the start of a burst, estimate the channel characteristics and frequency offset. We assume that these parameters are constant during the burst, because of the HiperLAN/2 standard requirements, the channels' coherence time (of about 10 ms) and the introduced phase noise by the oscillator [9].

Demodulation of data-OFDM symbols consists of four parts:

- frequency-offset correction
- 64-point FFT
- channel equalization
- QAM demodulation

1) Computational requirements:

synchronization/parameter estimation: Current research [9] focusses on synchronization and therefore the used algorithms are at the moment unknown. At this moment in time it is assumed that burst synchronization only is needed, so the required processing requirements will be far less than the processing costs of demodulation of data OFDM symbols.

frequency-offset correction: The frequency offset is detected by the synchronization part and corrected in the frequency-offset-correction part of the receiver. It requires one complex multiplication per sample. An OFDM symbol has a duration of 80 complex samples. The first 16 samples are the prefix, therefore only 64 samples have to corrected. The computation load is $N_s * 64 = 6.4$ million complex multiplications (25.6 million multiplications and 12.8 million additions) per second.

64-point FFT: The receiver has to perform a 64-point FFT every OFDM symbol. As a 16-bit receiver degrades the performance [10] we assume that the FFT has to be more accurate, namely 24 bit. According to [11], a P-points FFT requires $Plog_2(P)$ complex multiplications. So in our case the requirements are: 384 24-bit complex multiplications.

channel equalization: The equalizer has to "undo" the channel for the 48 data carriers. This requires 48 complex multiplications.

QAM demodulation: The largest constellation used is 64-QAM. A 64-QAM symbol has $2^3 = 8$ possible values for both the real and imaginary part. De-mapping can be implemented by generating an index for a table. Furthermore the borders must also be checked. So de-mapping requires 2 comparisons (border checking), 1 addition, 1 multiplication and 1 table lookup. An OFDM symbol has 48 data carriers and each QAM symbol requires two de-mapping operations.

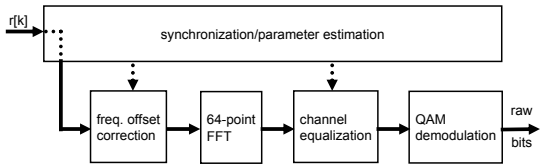

Fig. 3. OFDM receiver

stage	# million x	# million +	# million <	# bits
SRC	500	960	0	16 bit
synchronization	t.b.d.	t.b.d.	t.b.d.	16 bit
freq. offset corr.	25.6	12.8	0	16 bit
FFT	153,6	76.8	0	24 bit
channel equalization	19.2	9.6	0	16 bit
QAM-demodulation	9.6	9.6	19.2	16-bit

TABLE II
COMPUTATIONAL REQUIREMENTS FOR HIPERLAN/2 RECEPTION

Table II shows an overview of the estimated computational load for each part of the receiver.

III. FUNCTIONAL ARCHITECTURE OF A BLUETOOTH RECEIVER

Bluetooth uses GFSK as modulation technique. The symbol duration is 1μs and data is transmitted in time slots with a duration of 625μs. For estimating computational requirements, we assume maximal transfer rate. In this mode, Bluetooth uses a packet which spans 5 time slots and 1 time slot is used for uplink communication. Moreover we assume continuous transmission. (Of course, in realistic situations this is not the case.) For all parts we assume that 16-bit fixed point calculations are sufficient.

A. Channel selection

In Bluetooth mode the output of the analog front-end is also a 20 MHz complex signal containing 20 Bluetooth channels. Channel selection can be divided in three parts:

- Sample rate reduction (from 80 to 20 MSPS)
- Mixing of the wanted channel to baseband
- Removing adjacent channels

The first part, Sample rate reduction, is equal to the channel selection in HiperLAN/2 mode. The next step consists of mixing the wanted channel to baseband by multiplying the signal with a complex frequency. The final step is a low pass filter which eliminates all other channels.

1) Computational requirements:

Sample rate reduction: This part is equal to the channel selection in HiperLAN/2 mode, so symmetric FIR filters are assumed operating at 80 MSPS (at the input rate) with 25 taps and a decimation factor 4. The computational load is: $2 * \frac{25}{2} * \frac{80}{4} = 500$ million 16-bit multiplications and $2 * (25 - 1) * \frac{80}{4} = 960$ million 16-bit additions per second.

Mixing: Mixing requires one complex multiplication per sample. Mixing is only required during the reception of a packet (which is $\frac{5}{6}$ of the input sample rate of 20 MSPS).

Removing adjacent channels: This part is a low-pass filter. The used MAP receiver requires a symmetric 50-tap FIR for both the real and imaginary part. Furthermore the sample rate is reduced to 4 MSPS (4 samples per bit). The computational load is: $2*\frac{50}{2}*\frac{20}{5}*\frac{5}{6} = 166.7$ million 16-bit multiplications per second and $2*(50-1)*\frac{20}{5} = 326.7$ million 16-bit additions per second. These computational requirements can probably be scaled down, however further simulations are needed to verify this.

B. Demodulation

GFSK receivers use often a a so-called *FM discriminator* for demodulation [12]. The output of this FM discriminator are soft bits and a comparator is often used for hard bit decisions. The performance is not optimal but implementation is easy and low-power (AD conversion is performed by the comparator). The demodulator requires a real input signal at low-IF that does not map easily on the (complex) HiperLAN/2 demodulator. Furthermore the optimal channel selection filter is not defined by this demodulator. Therefore we have researched more advanced demodulators e.g. the MAP receiver. This receiver requires an orthogonal vector space which is given by the Laurent decomposition [13]. This Laurent decomposition describes the GFSK signal by a sum of linear, orthogonal, Pulse Amplitude-Modulated (PAM) waveforms.

C. Laurent decomposition

In [13] it has been shown that GFSK and in general Continous Phase Modulation (CPM) signals can be written as a sum of PAM waveforms. In many cases the signal power is concentrated in the first pulse, c_0. In the Bluetooth case [14], the first pulse contains about 99 % of the signal power. So, the GFSK signal can be approximated by using only this pulse (which simplifies the construction of the MAP receiver):

$$\tilde{r}(t, \boldsymbol{\alpha}) \approx \sum_n b_{0,n} c_0(t - nT) \tag{1}$$

where $b_{0,n}$ is the so-called *pseudo symbol* that is given by:

$$b_{0,n} = exp\{jh\pi[(\sum_{m=-\infty}^{n} \alpha_m)]\} \tag{2}$$

with α_m the m^{th} data bit and h the modulation index.

Moreover this first pulse, c_0, defines also the channel selection filter (the low pass filter).

D. MAP receiver

The MAP receiver is shown in Fig. 4.

The channel selection filter is a matched filter for the first Laurent waveform c_0. The output of the filter is an (optimal) estimation of $b_{0,n}$. This estimation has an optimal $\frac{E_b}{N_0}$ but suffer from Inter-Symbol Interference (ISI). An efficient search algorithm will be needed which determines the *optimal* path through the trellis diagram. For our MAP receiver we used the *Viterbi* algorithm with 2 states. This receiver requires (for the smallest modulation index) an $\frac{E_b}{N_0}$ of about 11 dB [14] for a BER of 10^{-3} (as is the BER required by the Bluetooth standard).

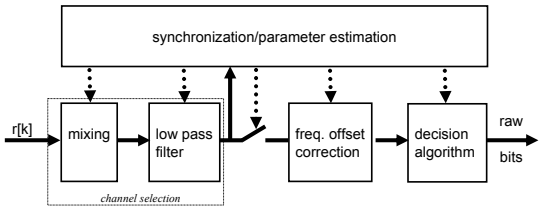

Fig. 4. MAP receiver

stage	# million x	# million +	# million <	# bits
SRC	500	960	0	16 bit
mixing	66.7	33.3	0	16 bit
low pass filter	166.7	326.7	0	24 bit
synchronization	t.b.d	t.b.d.	t.b.d.	16 bit
freq. offset corr.	3.3	1.67	0	16 bit
MAP receiver	29.9	21.6	2.5	16-bit

TABLE III
COMPUTATIONAL REQUIREMENTS FOR THE BLUETOOTH RECEPTION

synchronization/parameter estimation: Data is transmitted in bursts of maximal 3.125 ms (5 time slots), that starts with special a special code, the so-called *access code* [7]. The synchronization part can use this access code to detect the start of a burst, estimate the frequency offset and modulation index. Exact knowledge of the modulation index is needed for the MAP receiver because this value determines the states in the Viterbi algorithm. Estimation of the channel is not needed because the channel bandwidth is smaller than the *coherence bandwidth* [15] (of about 1 MHz). The synchronization part also determines the optimal sample moment and decimates the 4 MSPS to the symbol rate (of 1 MSPS).

1) Computational requirements:

synchronization/parameter estimation: Current research focusses on synchronization (and the combination with HiperLAN/2 synchronization) and therefore the used algorithms are at the moment unknown. We assume that burst synchronization only is needed and the required processing requirements will be about the same as for HiperLAN/2 synchronization/parameter estimation.

Frequency offset correction: The frequency offset is detected by the synchronization part and corrected in the frequency-offset-correction part of the receiver. It requires one complex multiplication per sample. The sample rate is $\frac{5}{6} = 0.83$ MSPS.

MAP receiver: The MAP receiver consists of a 2-state-Viterbi algorithm. This algorithm has to calculate for each state 2 branches and select the best branch. The state with the highest values determines the detected bit. Each branch requires 2 or 3 complex multiplications and in total the Viterbi algorithm requires 9 complex multiplications, 4 complex additions and 3 comparisons (36 multiplications, 26 additions and 3 comparisons). The Viterbi algorithm also operates at 0.83 MSPS.

Table III shows an overview of the estimated computational load for each part of the receiver.

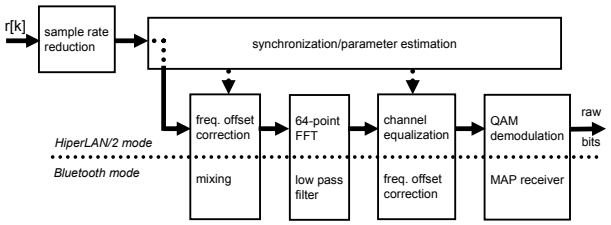

Fig. 5. A Bluetooth-enabled HiperLAN/2 receiver

IV. A BLUETOOTH-ENABLED HIPERLAN/2 RECEIVER

The Bluetooth-enabled HiperLAN/2 receiver is depicted in Fig. 5. For both receivers, the first step, sample rate reduction, is the same. In this step, the sample rate is reduced from a 80-MSPS to a 20-MSPS complex signal. The frequency offset correction of the the HiperLAN/2 receiver can be integrated with the mixing step of the Bluetooth receiver. Frequency offset correction is the same as mixing; both steps multiply the input signal with a complex carrier.

In the HiperLAN/2 receiver, the FFT has the highest computation requirements (see Table II), whereas, in Bluetooth mode the low-pass filter in the channel selection function requires most processing power (Table III). As filtering and FFT both incorporate multiplications and additions, it is possible to combine them. At this moment we assume that the computational requirements of the low pass filter can be scaled down to the requirements of the FFT without too much performance loss. Further simulations are needed to verify this. Low-pass filtering in the frequency domain is not an option because the *overlap-add* method [11] requires more computations[2]. Other combinations are now very straightforward. The HiperLAN/2 equalizer which incorporates complex multiplications can be combined with the Bluetooth frequency offset correction. Finally the MAP receiver can be combined with the QAM demodulator. In our current design of the HiperLAN/2 receiver we use a very simple demodulator. More complex QAM-demodulator have a soft-output and are integrated with the Forward-Error Correction (FEC) decoder that also contain a Viterbi-algorithm implementation.

The resulting partitioning of the Bluetooth receiver functions on the HiperLAN/2 receiver parts is very balanced both from a functional-partitioning perspective and from a computational-load perspective. (See Table II and Table III.)

V. CONCLUSIONS AND RECOMMENDATIONS

This paper presents a functional architecture of a Bluetooth-enabled HiperLAN/2 receiver. The Bluetooth standard is designed for low power and low cost receivers. Therefore a large part of the receiver is implemented in the analog domain which does not map on a digital OFDM receiver. In our project we used a MAP receiver for Bluetooth. This receiver has better and even optimal performance compared with commonly used Bluetooth receivers. Moreover this receiver can be mapped on the HiperLAN/2 receiver. In the proposed SDR receiver, channel selection of the Bluetooth receiver has been integrated with the frequency offset correction and FFT of the HiperLAN/2 receiver. Moreover the HiperLAN/2 QAM demodulator (and FEC decoder) can be combined with the Bluetooth MAP receiver.

Further research focusses on synchronization/parameter estimation for both standards. At this moment in time we assume that for both standards only burst synchronization is needed. The proposed SDR receiver can easily be adapted to other WLAN standards because other WLAN standards use the same frequency bands and modulation techniques.

[2]This method requires for this applications at least a 128-point FFT and a small 8 point IFFT (decimation and mixing can be performed after the FFT). So far more computations are required than "normal" filtering in the time domain.

ACKNOWLEDGMENT

We thank our colleagues from the IC-Design group for their work on the analog part of the front-end and for interesting discussions. This research is supported by the PROGram for Research on Embedded Systems & Software (PROGRESS) of the Dutch organization for Scientific Research NWO, the Dutch Ministry of Economic Affairs and the technology foundation STW.

REFERENCES

[1] "The bluetooth-hiperlan/2 sdr receiver project," http://sas.el.utwente.nl/home/SDR/.

[2] J. Mitola, *Software Radio Architecture: Object-Oriented Approaches to Wireless Systems Engineering*. Wiley, 2000.

[3] W. Tuttlebee, *Software Defined Radio: Origins, Drivers and International Perspectives*. John Wiley & Sons, 2002.

[4] V. Arkesteijn, E. Klumperink, and B. Nauta, "An analogue front-end architecture for software defined radio," 13^{th} *proRISC workshop*, November 2002.

[5] V. Arkesteijn, R. Schiphorst, F. Hoeksema, E. Klumperink, B. Nauta, and C. Slump, "A software-defined radio test-bed for wlan front ends," 3^{nd} *PROGRESS workshop on Embedded Systems and Software*, 2002.

[6] V. Arkesteijn, E. Klumperink, and B. Nauta, "An analogue front-end architecture for software defined radio," *MMSA2002*, December 2002.

[7] BluetoothSIG, "Specification of the bluetooth system - core," 'Bluetooth SIG', Technical Specification Version 1.1, February 2001.

[8] ETSI, "Broadband radio access networks (bran); hiperlan type 2; physical (phy) layer," ETSI, 650 Route des Lucioles - Sophia Antipolis, Valbonne - FRANCE, Technical Specification ETSI TS 101 475 V1.2.2 (2001-02), February 2001.

[9] F. Hoeksema, R. Schiphorst, and C. Slump, "Spectral weighting functions for single-symbol phase-noise specifications in ofdm systems," *To appear in the proceedings of the* 8^{th} *International OFDM-Workshop*, September 2003.

[10] L. van Hoesel, "Design and implementation of hiperlan/2 physical layer models for simulation purposes," Master's thesis, University of Twente (Department of Electrical Engineering), August 2002.

[11] A. Oppenheim and R. Schafer, *Discrete-Time Signal Processing*. Prentice Hall, Inc., 2002.

[12] R. Schiphorst, F. Hoeksema, and C. Slump, "Channel selection requirements for bluetooth receivers using a simple demodulation algorithm," 12^{nd} *proRISC workshop on Circuits, Systems and Signal Processing*, 2001.

[13] P. Laurent, "Exact and appropiate construction of digital phase modulateds by superposition of amplitude modulated pulses (amp)," *IEEE transactions on communications*, vol. 34, no. 2, pp. 150–160, February 1986.

[14] R. Schiphorst, F. Hoeksema, and C. Slump, "A (simplified) bluetooth maximum a posteriori probability (map) receiver," *Proceedings of IEEE SPAWC2003*, June 2003.

[15] T. Rappaport, *Wireless Communications*. Prentice Hall, Inc., 1996.